Architectural Requirements for a Multipurpose Natural Language Processor in the Clinical Environment

Carol Friedman Ph.D.[†], Stephen B. Johnson, Ph.D.[‡]
Bruce Forman, M.D.[‡], Justin Starren, M.D., M.S.[‡]

[†] Department of Computer Science, Queens College of CUNY and
Department of Medical Informatics, Columbia University, New York
[‡] Department of Medical Informatics, Columbia University, New York

A considerable amount of research has been concerned with the development of natural language systems to automate the encoding of clinical information that occurs in textual form. The task is very complex, and not many language processors are used routinely within clinical information systems. Those systems that are operational, have been implemented in narrow domains for particular applications. For a system to be truly useful, it should be designed so that it could be widely used within the clinical environment. This paper examines architectural requirements we have identified as being necessary for portability and describes the architecture of the system we developed. Our system was designed so that it could be used in different domains to serve a variety of applications. It has been integrated with the clinical information system at Columbia-Presbyterian Medical Center where it routinely encodes clinical information from radiological reports of patients.

INTRODUCTION

Many clinical information systems (CIS) include applications which require the use of controlled vocabularies. For example, alerting systems, such as the HELP system [1] and the CPMC system [2], use medical logic to query patient data which is represented as controlled terms. Diagnostic systems, such as QMR [3] and DX*plain* [4], use controlled terminology to obtain clinical information for making diagnoses. Information retrieval applications [5, 6, 7] use the UMLS [8] or MeSH [9] vocabularies in order to access the medical literature.

Natural language processing systems have been used in limited domains [10, 11, 12, 13, 14, 15, 16, 17] to automatically obtain controlled vocabulary data from clinical reports so that the data could be used by subsequent automated applications. Since clinical information systems typically contain a wealth of online clinical data in textual form, a natural language processor could significantly enhance the functionality of a CIS by automatically obtaining controlled terms from the clinical reports, thereby supplying the automated applications within the CIS with a more complete data set. For a language processing system to be truly useful, it should be capable of functioning in different domains within the clinical environment.

We have developed a natural language system, called MedLEE (an acroymn for **Med**ical **L**anguage **E**xtraction and **E**ncoding System), which has been integrated with the CIS at Columbia-Presbyterian Medical Center (CPMC) to routinely obtain coded data. Currently the data is being used by the alerting component of the CIS. The initial application, which is described in more detail in [11, 18], is the extraction, encoding, and uploading (to the coded clinical database) of information occurring in radiological reports of patients at CPMC. Although radiology is the initial application, MedLEE is also intended to be used for different domains and different applications. We therefore developed an architecture which will facilitate porting the processor.

A number of articles describe the design of individual natural language systems. However few focus on architectural requirements needed to achieve generality. This paper discusses design requirements that are necessary for a general natural language processor within the clinical environment. This paper also outlines the most recent architectural design of MedLEE, and discusses how it meets the requirements we set forth.

RELATED WORK

The Linguistic String Project (LSP) [19, 10] under the leadership of Sager was a pioneer in the development of a medical language processing system, and the LSP system is still the most comprehensive of such systems. It has been applied to a variety of domains, such as discharge summaries, radiology, asthma, rheumatoid arthritis, and pharmocology (literature). The system produces a structured output where the terms are standardized but not encoded. Recent work [20] has investigated mapping the output to SNOMED codes. The LSP system has separate knowledge components specifying syntactic and semantic properties.

Other systems that have been implemented were applied to only one or two domains. These generally rely more heavily on semantic information and heuristics. A coding system that produces SNOMED codes [21] has been developed for pathology. This system uses a word barrier technique that partitions the text sentence to isolate medical terms that are candidates for translation to SNOMED codes. Other systems,

0195-4210/95/$5.00

which also use a partitioning technique have been applied to radiology [14, 12], physical examination summaries [22], gastro-intestinal surgery [13], and echocardiography [23]. In these systems, the sentences are first partitioned into phrases and are subsequently processed further using either semantic or syntactic methods. A heuristic slot filling technique is then used to obtain the final encoded form.

REQUIREMENTS

A detailed description of the architecture of our system has been discussed previously [11]. In this section we will discuss our design requirements and also our design, and explain how this infrastructure supports portability.

Separate Processor and Knowledge

Typically, natural language processors incorporate a large body of different types of knowledge. It is unlikely that any processor can anticipate all the knowledge that will be required for any domain or task. Therefore a processor should maintain a rigid separation between the *knowledge* components and the *control engine* component. This isolates the general computational algorithm from the domain specific elements so that the processor always remains the same but is driven by domain specific elements which are expected to be changed. This requirement facilitates the development of new applications, because an application may be developed rather quickly for a very limited use, and then be expanded in incremental stages by suitable augmentations of the knowledge sources only.

Separate Components of Knowledge

There are usually different types of knowledge required for natural language processing. These different types should be maintained as independent components which are in the form of tables. This reduces the overall complexity of the system, and makes it both more manageable and portable. This type of infrastructure facilitates the development of new applications because it allows implementers with different types of expertise to work only on those components that are relevant to them. When a knowledge component is in the form of a table, it is much easier to access, understand, and modify than embedded programming code.

The four different knowledge components that we utilize are:

1. Lexicon: This component identifies and categorizes single-words and multi-word phrases that occur in the text. It also specifies target output forms. In our system, the lexical entries generally have broad semantic categories. For example, *lung* is categorized as a body location, whereas *mass* is categorized as a finding.

2. A Language Model: This contains knowledge about the structure of the language in the domain and a formal representational schema which describes the target structure. In our design, the structure of the source language is specified in a context-free semantic grammar which defines the well-formed semantic (and some syntactic) structures of the domain. In addition, each definition also specifies a target structure and describes how to map the components of the structure into the target form. For example, the simple structure **finding in bodyloc** (i.e. *mass in lung*), is interpreted as a finding qualified by a body location, and a target structure is generated accordingly. The structure will have a **finding** relation with the value **mass**, and a **bodyloc modifier** relation with the value **lung**. A detailed description of the formal representation of clinical information in our system is discussed further in [24].

Defining the semantic patterns along with the corresponding target structure combines the functionality of parsing and slot filling, thereby eliminating the need for heuristic slot filling procedures, which are typically embedded in programming code and therefore are likely to require re-programming when changing domains. Specifying target structures for semantic patterns also provides a way to explicitly represent the semantic relations among the components of a structure. For example, in a structure which consists of the components **neg change finding** (i.e. *no increased swelling*), we can specify that **neg** is related to **change** rather than to **finding**.

3. Links to a Controlled Vocabulary: This component establishes correspondences between textual terms and controlled vocabulary terms. It also makes explicit commonly omitted domain-specific information, and it adjusts the granularity of the controlled terms to the appropriate level for an application.

The output of a processor should ultimately consist of terms which correspond to a controlled vocabulary, yet in order to achieve flexibility, it is advantageous for intermediate output to be generated that is initially independent of a particular vocabulary. Some vocabularies are more suitable for certain applications than others. For example, CPT4 [25] is a standard for reimbursement for procedures, SNOMED [26] was designed to represent comprehensive clinical information, and MeSH and the UMLS are useful for indexing the medical literature. In addition, in other automated medical systems, unique standalone controlled vocabularies have been developed that are utilized for particular applications. By maintaining a separate knowledge base which links canonical textual terms to a controlled vocabulary, a language processor can serve multiple applications by interchanging this component when necessary.

At CPMC the controlled vocabulary currently contains over 40,000 terms which are managed in the Medical Entities Dictionary (MED) [27]. Our processor first generates an intermediate target form that contains terms that are specified in the lexicon, and therefore, that are based solely on lexical information. For example, the target form of *increase* and *increasing* is the canonical form **increase**. At a later stage in processing, the linking component is used to map the lexically-based target terms to terms associated with the controlled vocabulary. For example *adenopathy* is not in the MED but is linked to the term **enlargement of lymph nodes**, which is in the MED.

The linking component is also used for adjusting the granularity of the final output to the appropriate level for an application. For example, an alerting application being planned at CPMC will use encoded data from mammography reports to detect suspicious findings in order to automate the monitoring of follow-up care. For this application, the specific location of the

finding within the breast is not needed. This means that many specific locations such as *left outer quadrant*, *left upper quadrant*, *left inner quadrant*, etc., can be associated with a singular controlled vocabulary term **breast**. Mapping many specific textual terms to one controlled term will significantly reduce the vocabulary effort for this particular application without damaging functionality. Having fewer controlled terms is also likely to have a positive effect on the accuracy of subsequent queries associated with the information because fewer terms need to be included in the query.

In other situations, a finer granularity may be desired. The linking component is also used to fine tune terms within a domain, and to explicitly represent information which is often implicit in the reports. Terms are frequently omitted within specific domains because they are typically inferred by domain experts. For example, when encoding mammography reports, the phrase *cystic disease* maps to the MED term **diffuse cystic mastopathy**, but in the chest x-ray domain, it maps to the MED term **cystic disease**. Similarly, *decreased volume* in a chest x-ray maps to **loss of lung volume**, but could be associated with a different term in another domain.

4. Compositional Component: Words that are part of multi-word phrases frequently get separated in textual reports. This component provides knowledge that models the compositionality of multi-word phrases so that if the words of a phrase are separated in a sentence, the processor will be able to combine them into one term, thereby enabling mapping to a controlled vocabulary term at a later phase. For example, *elevated diaphragm* is a finding that occurs in chest x-rays, but in a report it may be expressed as *elevation of the diaphragm* or *the diaphragm appears to be moderately elevated.* Having a model of compositionality eliminates the need for having to introduce an explosion of lexical variants in the system or for requiring procedural matching algorithms that would otherwise be needed in order to fix up the target form.

Minimize Use of Inferential Knowledge

Some processing systems incorporate inferential medical knowledge that is not associated with terminology. For example, some systems infer diagnoses based on findings in the report, and others identify whether a follow up is required because of the presence of certain findings. We exclude this type of knowledge because it significantly increases the complexity of the processor and ties it to very specific domains and applications. In addition, the data needed for inferential purposes may be incomplete in a particular report but it may be present in another informational source in the CIS. Because a substantial amount of effort has already been expended independently of natural language processing in the development of clinical applications such as alerting and decision support systems, we believe it is more effective and more efficient to minimize the complexity of the processor by including only those knowledge components which are essential for the task of language processing. Functionality is gained rather than lost by this division because automated clinical decision support and alerting procedures could be used subsequent to the encoding. These procedures should be more effective because they would have access all the encoded patient data.

A Flexible Interface

A language processor should have a flexible interface so that only one version need be maintained within the CIS for all different types of applications. This entails that the input to the processor be in a form which is common to all reports (i.e. sentences). Since the format of different types of textual reports typically differ, this necessitates that other higher level applications process the overall structure of the reports. In addition, different types of output are typically required for different applications. The language processor should generate a structure which is well-defined and easy to manipulate.

Figure 1 shows how MedLEE interfaces with applications in the CIS at CPMC. It functions as a natural

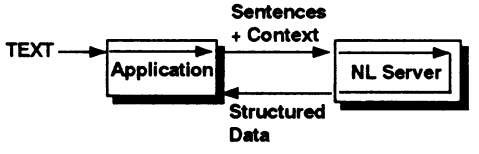

Figure 1 - Natural Language Server in CIS

language server (NL Server) which receives a set of sentences (along with parameters specifying contextual information such as the name of the examination, the section of the report, and the type of output desired) from the application. It processes the input and returns structured data in the desired format to the application. In this architecture, it is the function of the application to find the textual sections that are to be processed, to pass the sentences in the section to MedLEE, and to handle the structured output when it has been returned.

The original design of MedLEE as described in [11] included a preprocessing component which contained a detailed description of the structure of the overall report. One of its functions was to identify the different sections of the report and to process them accordingly. This design tied the language processor closely to a particular report format and required that different versions of the processor be maintained for different types of report formats. The design was also inefficient because it used natural language processing techniques (which are more complex than those needed for straightforward text manipulation tasks) for all the sections of the report. However, a majority of the sections were naturally formatted and therefore could be handled more efficiently using simpler techniques. Presently, a Perl [28] script is used to process the overall report (which is in HL7 format) because it is ideal for inter process communication and text manipulation. The Perl script finds the sentences in the relevant sections and calls MedLEE using the appropriate parameters. MedLEE computes and returns the encoded data (which, in this case, is in HL7 format). The script then creates a new HL7 message and stores it in a directory so that the encoded data is ready for uploading using the standard CIS upload interface.

The new design provides a simple interface for using MedLEE for different applications. At CPMC, a number of *reformatting* processes already exist that are utilized when clinical reports are received by the CIS from

different departmental systems. Typically a report is received in HL7 [29] format but still must be reformatted so that it is consistent with the CIS architectural requirements. With this infrastructure in place, language processing could easily be embedded in the upload process by modifying the reformatting programs to call MedLEE to obtain encoded data as part of the standard upload application. Presently the encoding of radiology reports is not embedded in the reformatting process but is performed as a separate process.

DISCUSSION

Natural language systems are difficult to develop and maintain because the task is complex and knowledge intensive. Systems containing less knowledge can be developed more quickly and they are easier to maintain, but they still require a substantial amount of effort to develop. A few are actually used in an operational mode on a routine basis, but often they apply only to a narrow domain for a specific application. It may be not possible for some to be used for different applications, whereas others may be ported but with great difficulty because of their underlying architectures.

We have described an architecture which is necessary for portability. However, other additional issues are also relevant. When moving to a new domain, the lexicon and language model must be adapted. This entails that new words and multi-word phrases in the domain must be discovered and added to the lexicon. The discovery of multi-word phrases is not straightforward and tools are needed to facilitate the process. The CLARIT system [30] uses natural language techniques to automatically create a ranked list of terms for the domain by using a large sample corpus. We have also developed an automated tool that uses statistical techniques to propose new multi-word phrases from a large corpus. However, categorization of the phrases and also the words must be done manually. Automated tools developed for this purpose will greatly facilitate lexical work.

The component which establishes links from the processor to the vocabulary requires medical expertise to manually examine the controlled vocabulary and determine associations between the lexically based target terms and the controlled vocabulary terms. Automated tools to aide in this process are also needed. Currently, the MED has a browser to help navigate the hierarchy of terms, but this is still a manual process.

The language model was also designed manually. It is basically general within the clinical domain because it consists primarily of findings and modifiers, but there are some constructs which are domain-specific. When moving to a new domain, it is likely that the language model will have to be adjusted. We do not anticipate this will be a major undertaking as long as we remain within the clinical domain, but this is still an open question.

To summarize, the requirements we identified that are necessary for portability are: 1) separating knowledge components from the processing engine, 2) maintaining separate knowledge components according to functionality, 3) maintaining knowledge in the form of declarative tables rather than embedded in programming code, 4) developing a uniform and flexible interface for the processor, and 5) minimizing inferential knowledge. In addition we suggested that automated tools should be used to develop new or expanded knowledge for a domain.

It is well-known that natural language processing is enormously difficult, but its potential is also enormous. It has been shown to be useful within limited domains, but no single processor that is operational has applied the technology to many domains or applications. We believe that it is important to analyze the underlying infrastructure for portability because only then will it be possible for language processing to become a powerful technology within medical informatics.

Acknowledgements

This publication was supported in part by grants LM05397 and LM07079 from the National Library of Medicine and by the Columbia CAT in High Performance Computing and Communications in Healthcare, a New York State Center for Advanced Technology supported by the New York State Science and Technology Foundation.

References

[1] T.A. Pryor, R.M. Gardner, P.D. Clayton, and H.R. Warner. The HELP system. *Journal of Medical Systems*, 7, 1983.

[2] Hripscak G, Cimino JJ, Johnson SB, and Clayton PD. The Columbia-Presbyterian Medical Center decision-support system as a model for implementing the Arden Syntax. In Clayton PD, editor, *Fifteenth Annual Symposium on Computer Applications in Medical Care*, McGraw-Hill, New York, 1992.

[3] Miller RA, McNeil MA, Challinor SM, Masarie FF, and Myers JD. The INTERNIST I Quick Medical Reference Project - status report. *West. J. Med.*, 145, Dec 1986.

[4] Barnett GO, Cimino JJ, Hupp HA, and Hoffer EP. DXplain. an evolving diagnostic decision-support system. *JAMA*, 258(1), 1987.

[5] Rindflesh TC and Aronson AR. Semantic processing in information retrieval. In Safran C, editor, *Proceedings of the Seventeenth Annual Symposium on Computer Applications in Medical Care*, McGraw-Hill, New York, 1994. 1993 Nov 1-3; Washington,(DC).

[6] Hersh WR, Hickam DH, and Leone TJ. Words, concepts, or both: optimal indexing units for automated information retrieval. In Frisse ME, editor, *Proceedings of the Sixteenth Annual Symposium on Computer Applications in Medical Care*, McGraw-Hill, New York, 1992.

[7] Miller RA, Gieszczykiewicz FM, Vries JK, and Cooper GF. CHARTLINE: providing bibliographic relevants to patient charts using the UMLS Metathesaurus knowledge sources. In Frisse ME,

editor, *Proceedings of the Sixteenth Annual Symposium on Computer Applications in Medical Care*, McGraw-Hill, New York:, 1992.

[8] Lindberg DAB, Humpreys BL, and MCCray AT. The Unified Medical Language System. *Methods of Information in Medicine*, 32, 1993.

[9] *Medical Subject Headings - Annotated Alphabetical List.* Technical Report, National Library of Medicine, published annually.

[10] Sager N, Lyman M, Bucknall C, Nhan N, and Tick LJ. Natural language processing and the representation of clinical data. *Journal of the American Medical Informatics Association*, 1(2):142–160, 1994.

[11] Friedman C, Alderson P, Austin J, Cimino JJ, and Johnson SB. A general natural language text processor for clinical radiology. *Journal of American Medical Informatics Association*, 1(2):161–174, March 1994.

[12] Zingmond D and Lenert LA. Monitoring free-text data using medical language processing. *Computers and Biomedical Research*, 26, 1993.

[13] Baud RH, Rassinoux A-M, and Scherrer J-R. Natural language processing and semantical representation of medical texts. *Methods of Information in Medicine*, 31(2), 1992.

[14] Ranum D. Knowledge based understanding of radiology text. In Greenes RA, editor, *Proceedings of the Twelfth Annual Symposium on Computer Applications in Medical Care*, pages 141–145, IEEE Computer Society Press, Washington, D.C., 1988.

[15] Moore GW and Berman JJ. Automatic (SNOMED) coding. In Ozbold JG, editor, *Proceedings of Eighteenth Annual Symposium on Computer Applications in Medical Care*, Hanley and Belfus Inc, Philadelphia, 1994.

[16] Wingert F, Rothwell D, and Cote RA. Automated indexing into SNOMED and ICD. In *Computerized Natural Medical Language Processing for Knowledge Representation*, Elsevier Science, The Netherlands, 1989.

[17] Scherrer J-R, Assimacopoulos A, and Griesser V. The DIOGENE Geneva experience with medical interactive encoding: the role of a micro-glossary. In *Computerized Natural Medical Language Processing for Knowledge Representation*, Elsevier Science, The Netherlands, 1989.

[18] Friedman C, Hripcsak G, DuMouchel W, Johnson SB, and Clayton PD. Natural language processing in an operational clinical information system. *Journal of Natural Language Engineering*, 1995. in press.

[19] Sager N, Friedman C, and Lyman MS et al. *Medical Language Processing: Computer Management of Narrative Data.* Addison-Wesley, Reading, MA, 1987.

[20] Sager N, Luman M, Nhan NT, and Tick LJ. Automatic encoding into SNOMED III: a preliminary investigation. In Ozbold JG, editor, *Proceedings of the Eighteenth Annual Symposium on Computer Applications in Medical Care*, Hanley and Belfus Inc, Philadelphia, 1994.

[21] Moore GW and Berman JJ. Performance analysis of manual and automated synstematized nomenclature of medicine (SNOMED) coding. *Am J Clin Pathol*, 101, 1994.

[22] Lin R, Lenert LA, Middleton B, and Shiffman S. A free-text processing system to capture physical findings: canonical phrase identification system (CAPIS). In Clayton PD, editor, *Proceedings of the Fifteenth Annual Symposium on Computer Applications in Medical Care*, McGraw-Hill, New York:, 1992.

[23] Canfield K, Bray B, Huff S, and Warner H. Database capture of natural language echocardiology reports: a UMLS approach. In *Proceedings of the Fourteenth Annual Symposium on Computer Applications in Medical Care*, IEEE Computer Society Press, Los Alamitos, CA, 1990.

[24] Friedman C, Cimino JJ, and Johnson SB. A schema for representing medical language. *Journal of American Medical Informatics Association*, 1(3):233–248, May 1994.

[25] Clauser SB and eds Fanta CM. *Current Procedural Terminology, Fourth Edition - CPT4.* Technical Report, AMA, Chicago, 1984.

[26] *The Systematized Nomenclature of Medicine: SNOMED International.* Technical Report, College of American Pathologists, 1993.

[27] Cimino JJ, Clayton PD, Hripcsak G, and Johnson SB. Knowledge-based approaches to the maintenance of a large controlled medical terminology. *J Am Med Informatics Assoc*, 1(1), 1994.

[28] Wall L and Schwartz RL. *Programming Perl.* O'Reilly & Associates, Inc, Sebastopol, CA, 1991.

[29] *Health Level Seven.* 1990. Version 2.1.

[30] Evans DA, Hersh WR, Monarch IA, Lefferts RG, and Handerson SK. Automatic indexing of abstracts via natural-language processing using a simple thesaurus. *Med Decis Making*, 11(suppl), 1991.